Aspect Based Sentiment Analysis

Sanjay Ramachandran
sramac22@uic.edu
UIN: 671035289

ABSTRACT

Aspect based sentiment analysis involves predicting the sentiment of a target aspect in the given sentence. A sentence can be made of multiple aspects with its own sentiment score. This report includes an analysis of Gadget reviews and Restaurant reviews using statistical and deep learning models that takes a sentence along with the target aspect as the input and outputs a prediction of the sentiment respective to that aspect. The prediction can be 1(positive), 0(neutral) or -1(negative). The report also talks about a few pre-processing steps, the features that were built for the statistical models and an evaluation of the different models using accuracy, precision, recall and f1 scores. The evaluation is done using 10-fold cross validation and the average of scores across the 10-folds is saved for comparison.

INTRODUCTION

Sentiment Analysis is the process of analyzing a given sentence, finding key words that express sentiments and predicting the sentiment that is being expressed in the sentence as a whole. The usual sentiments include a positive sentiment, negative sentiment or a neutral sentiment. Instead of predicting the sentiment of the entire sentence, as explained in [1], a sentence can be made of multiple aspects with its own sentiment orientation. Models used for sentence sentiment classification do not work well in such cases. For eg., consider the review, 'The phone is good but the sound is bad'. There are two aspects in this review, viz. The phone and The sound, with positive and negative sentiments respectively. This kind of analysis has gained more light in the recent times since people consider reviews very seriously before purchasing a product and the reviews are not always simple and talks about multiple aspects of the product/service in a single post.

In this project report, multiple models have been analyzed and evaluated with the task of predicting the sentiment of the target aspect in a given sentence. The training data includes reviews for two specific domains, viz. Technology and Restaurant. The number of target sentiments is three (positive, neutral, negative). Pre-processing and feature engineering are performed before training the models. Tf-Idf vectors built using the training data and Word embeddings from Google's pre-trained word2vec are also used as part of the features. The model validation is performed using 10-fold cross validation. The evaluation measures used are average accuracy, average precision, recall and f-score.

DATA PRE-PROCESSING

The pre-processing steps that were performed are:

- Punctuations and stop words removal
- Stop words that indicate negation or complement like but, not, doesn't, cannot, etc. are retained in the sentences
- Numbers to '#' (to be used with Google's word2vec). For tf-idf vectors, numbers are removed
- [comma] are replaced with ','
- Aspect terms, with multiple words, in the sentences are concatenated with underscore for creating the tf-idf vectors. For word2vec, aspects are considered as phrases
- Sentences are clipped to a maximum length of 20 words and all words are lowercased. If a sentence has less than 20 words, then junk characters are padded in the end
- The target classes for the training data are extracted and saved separately

FEATURES

The following features were engineered for the statistical models and a subset of them is used in the models:

- The tokens in the sentences are converted to a 300-dimensional word vectors using the word2vec model
- **Attention weights** – Attention weights are calculated using similarity measures between the aspect and the tokens. The vectors created above are either weighted based on the attention scores or the reweighted vectors are concatenated with the original vectors resulting in 600-dimensional vectors
- For statistical models, an additional representation of words is used. Tf-idf vectors were built using the 1,2,3-grams in the training data
- **Context based lexicon scores** – The lexicons downloaded from [1] are used to check if a word belongs to the positive or negative lexicon. Words that fall inside a context window of 5 are extracted and a cumulative lexicon score is calculated
- **Word Importance** - Distance between nodes in parse tree – A dependency parse tree is created using [2]. It is then transformed to a graph. The importance of each word in the sentence is calculated as the reciprocal of the shortest path distance from the word to the aspect term in the graph.
- **Weighted lexicon score** – Instead of using a weight of one for all the words while calculating lexicon score, the importance calculated in the previous step is used as the weights and the product terms are summed up to give the lexicon score
- **Sentence Polarity** – The polarity of all the tokens in the training example is summed up to give the sentence polarity score

MODELS

A number of statistical, ensemble models were built for this task. A convolutional neural network was also built for trying out deep learning models. Multiple subsets of features developed above were given as inputs to the models.

- Linear Multi Class Support Vector Classifier (Linear SVC) – w and w/o class weights
- Gaussian Naïve Bayes (NB) Classifier
- Random Forest Classifier (RFC) with 100 estimators
- Convolutional Neural Networks (CNN)
    - An attention layer is applied to the input word vectors
    - 3 parallel convolutional layers are used with 200 filters each and with zero padding
    - Filter sizes are 2, 3 and 4 corresponding to learning features using 2-grams, 3-grams and 4-grams
    - The convolutional layers use ReLU activations
    - Each convolutional layer is followed by a Max-Pooling layer to capture the most important feature in the feature maps produced by that convolution
    - A dropout layer is added with threshold=0.5
    - This is followed by a fully connected output layer with softmax activations and l2 regularizer
    - The model is trained using Adam and categorical cross-entropy loss function

EXPERIMENT RESULTS

The different subsets of features are:

A) Google word2vec + attention
B) Google word2vec + Word Importance
C) Tf-idf vectors + context based lexicon scores + sentence polarity
D) Tf-idf vectors + weighted lexicon scores + sentence polarity

Technology domain:

	Positive			Neutral			Negative			
	P	R	F1	P	R	F1	P	R	F1	Accuracy
Gaussian NB + D	0.7606	0.7062	0.7311	0.4322	0.5804	0.4943	0.7265	0.6487	0.6831	0.6597
RFC + D	0.7885	0.8053	0.7965	0.6545	0.2630	0.3728	0.6696	0.8661	0.7552	0.7202
Linear SVC + D	0.7825	0.8014	0.7913	0.6095	0.4043	0.4831	0.7177	0.8244	0.7667	0.7310
Linear SVC + A	0.7469	0.7983	0.7717	0.6	0.2673	0.3699	0.6609	0.8036	0.7253	0.6947
Linear SVC + B	0.7587	0.8196	0.7874	0.6268	0.3021	0.4065	0.6849	0.8083	0.7411	0.7124
CNN + A	0.6945	0.7230	0.7085	0.3780	0.3369	0.3563	0.6235	0.6306	0.6271	0.6112
Gaussian NB + C	0.7539	0.7031	0.7269	0.4478	0.5826	0.5045	0.7288	0.6651	0.6941	0.6649
RFC + C	0.7837	0.8074	0.7939	0.6684	0.3239	0.4347	0.6969	0.8532	0.7654	0.7284
Linear SVC + C	**0.8142**	**0.7871**	**0.7995**	**0.5960**	**0.5065**	**0.5445**	**0.7343**	**0.8152**	**0.7722**	**0.7418**

Restaurant domain:

	Positive			Neutral			Negative			
	P	R	F1	P	R	F1	P	R	F1	Accuracy
Gaussian NB + D	0.8294	0.7153	0.7679	0.3460	0.5308	0.4186	0.5248	0.4967	0.5099	0.6340
RFC + D	0.7374	0.9385	0.8259	0.5479	0.2794	0.3688	0.6779	0.4384	0.5314	0.7109
Linear SVC + D	**0.8005**	**0.8932**	**0.8443**	**0.5305**	**0.3523**	**0.4222**	**0.6237**	**0.5925**	**0.6066**	**0.7309**
Linear SVC + A	0.7127	0.9219	0.8039	0.4777	0.1358	0.2115	0.5826	0.4509	0.5084	0.6785
Linear SVC + B	0.7350	0.9426	0.8259	0.5894	0.1705	0.2635	0.6110	0.4856	0.5394	0.7048
CNN + A	0.604	0.9953	0.7518	0.5666	0.0157	0.0304	0.2388	0.0148	0.0277	0.6041
Gaussian NB + C	0.8299	0.7079	0.7637	0.3497	0.5401	0.4235	0.5239	0.5043	0.5133	0.6329
RFC + C	0.7624	0.9112	0.8302	0.5251	0.3272	0.4006	0.6577	0.5057	0.5710	0.7179
Linear SVC + C	0.8360	0.8299	0.8328	0.4794	0.4626	0.4702	0.5942	0.6186	0.6052	0.7181

CONCLUSION

It is clear from the experiment results that feature engineering is a crucial part when it comes to text classification. CNN can still be tuned to perform better with more layers, filters and reworking the weighting layer. For the laptop domain, Linear SVC with the feature set C performs better than all other models. But for the restaurant domain, Linear SVC with the feature set D outperforms others. So it is safe to say that feature sets C and D help in structuring the input better and capturing the important features in the text.

Some future work can be:

- Implementing resampling techniques. There is a class imbalance issue in the dataset. That's the reason why neutral class has lower F1 score than the other two classes. Resampling will definitely improve precision, recall and f1 scores
- Building better neural networks using Long Short Term Memory and Recurrent Neural Networks to capture the features around the aspect term
- Updating word embeddings to better suit to the domain's lexicons

REFERENCES

[1] Bing Liu and Lei Zhang. 2012. A Survey of Opinion Mining and Sentiment Analysis. Mining Text Data, (Chapter 13):415–463.

[2] Spacy - https://spacy.io/

[3] ABSA using Gated CNN - https://arxiv.org/pdf/1805.07043.pdf

[4] ABSA using CNN and Attention layer - https://arxiv.org/ftp/arxiv/papers/1807/1807.01704.pdf

[5] ABSA using SVM and feature engineering - http://www.aclweb.org/anthology/S14-2076